Express Mail Label No. EV303487951US

# MEMORY MODULE WITH TESTING LOGIC

## BACKGROUND

**[0001]** Computer systems, for example home computers or high-end computers operated as servers, may utilize memory to store data. The memory may comprise one or more memory modules that are connected together via a memory bus. The memory bus may utilize a signaling convention that facilitates the transfer of data between devices connected to the bus. The devices may comprise memory devices, data converters, and remote input/output ports.

**[0002]** To ensure that the memory associated with a computer system is functioning properly, an error detection mechanism within the computer system's chipset may monitor the memory and detect memory errors. When a memory error is detected, an error handling procedure may correct and/or report the memory error.

**[0003]** To validate and test the error detection mechanism, errors may be artificially injected into the chipset or manually forced at memory level. As memory bus speeds have increased, it may be difficult to reliably inject errors for the purpose of testing and validating the error detection mechanism. Furthermore, artificially injecting errors into chipset may adversely affect system validation.

## SUMMARY

**[0004]** The problems noted above may be solved in large part by a memory module with testing logic. One exemplary embodiment may be a memory module that comprises a plurality of memory circuits, at least one serial presence detect (SPD) memory circuit, and a plurality of data lines that transfer data to and from the plurality of memory circuits. The memory module may further comprise a

testing logic that utilizes data stored in the SPD memory circuit to inject a memory error into one or more of the plurality data lines.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0005]** For a detailed description of the embodiments of the invention, reference will now be made to the accompanying drawings in which:

**[0006]** Figure 1 illustrates a computer system constructed in accordance with embodiments of the invention;

**[0007]** Figure 2 illustrates a memory module constructed in accordance with embodiments of the invention;

**[0008]** Figure 3 illustrates a flow diagram of an error injection procedure in accordance with embodiments of the invention; and

**[0009]** Figure 4 illustrates the testing logic of Figure 2 in accordance with embodiments of the invention.

## NOTATION AND NOMENCLATURE

**[0010]** Certain terms are used throughout the following description and claims to refer to particular system components. As one skilled in the art will appreciate, computer companies may refer to a component by different names. This document does not intend to distinguish between components that differ in name but not function.

**[0011]** In the following discussion and in the claims, the terms "including" and "comprising" are used in an open-ended fashion, and thus should be interpreted to mean "including, but not limited to ..." Also, the verb "couple" or "couples" is intended to mean either an indirect or direct connection. Thus, if a first device couples to a second device, that connection may be through a direct connection, or through an indirect connection via other devices and connections.

## DETAILED DESCRIPTION

**[0012]** The following discussion is directed to various embodiments of the invention. The embodiments disclosed should not be interpreted, or otherwise used, as limiting the scope of the disclosure unless otherwise specified. In addition, one skilled in the art will understand that the following description has broad application, and the discussion of any embodiment is meant only to be

exemplary of that embodiment, and not intended to intimate that the scope of the disclosure is limited to that embodiment.

**[0013]** Figure 1 illustrates an exemplary computer system constructed in accordance with embodiments of the invention. System 100 may be any type of computer system, such as a laptop computer, a personal computer, or stand-alone computer operated as a server. The system 100 may comprise a single central processing unit (CPU) 102, as illustrated in Figure 1, or may comprise a plurality of CPUs arranged in a configuration where parallel computing may take place. The CPU 102 may couple to a memory controller 104 that manages a memory 106.

**[0014]** The memory 106 may comprise one or more memory slots, each slot designed to interface with a memory module, such as a dual inline memory module (DIMM). Although any number of memory slots may be used, eight memory slots 108 – 122 are illustrated in the memory 106. Each memory slot 108 – 122 may interface with a memory module (not specifically shown in Figure 1) that may comprise any type of memory circuit, such as double data rate (DDR), fast page mode (FPM), and extended data out (EDO). The memory slots 108 – 122 may be coupled together and to the memory controller 104 via a memory bus (not shown) that facilitates the transfer of data between the memory slots 108 – 122 and the memory controller 104. The CPU 102, the memory controller 104, and the memory 106 may be associated with a chipset that provides a control and communication mechanism for the system 100. The chipset may comprise the memory slots 108 – 122 and any other hardware, such as registers, sockets, buffer, and serializers, required to operate the system 100.

**[0015]** Referring now to Figure 2, an exemplary memory module is shown in accordance with embodiments of the invention. The memory module 200 may comprise one or more memory circuits that are capable of storing data. Although any number of memory circuits may be used, eight memory circuits 202 – 216 are illustrated in the memory module 200. In addition, the memory module 200 may comprise a serial presence detect (SPD) 218 memory. The SPD 218 memory may be any type of non-volatile memory, such as electrically erasable programmable read-only memory (EEPROM), that stores data identifying the type

of memory module and various memory organization and timing parameters associated with the memory module 200. For example, the SPD 218 memory may store data describing the column access strobe (CAS) latency and the rank (*i.e.,* single or dual rank) of the memory module 200.

**[0016]** Data may be transferred to and from the memory circuits 202 – 216 via a plurality of data lines (not specifically shown). Each data line may couple to one or more input/output (I/O) pins 222 that interface the memory module 200 with a memory slot 108 – 122 (Figure 1). For example, a DIMM may possess 168 total I/O pins, 64 of which are coupled to distinct data lines. The pins not coupled to a data line may be used for various signals, such as address signals, power signals, ground signals, clock signals, and write strobe signals.

**[0017]** In accordance with embodiments of the invention, testing logic in the memory module 200 may inject memory errors into the chipset associated with the memory module by applying a bias voltage on one or more of the data lines. The testing logic 220 may couple to one or more of the data lines and be capable of injecting a memory error into the chipset. The testing logic 220 may be a programming logic array (PLA), a programmable logic device (PLD), or any other logical component capable of applying a bias voltage on a data line in the memory module 200. In addition, the testing logic 220 may couple to the SPD 218 memory and one or more I/O pins 222. The SPD 218 memory and one or more I/O pins 222 may act as inputs for the testing logic 220 and may provide data, such as CAS latency and rank, that indicates when data lines should be electrically biased by the testing logic 220.

**[0018]** Referring now to Figure 3, an exemplary error injection procedure is shown in accordance with embodiments of the invention. An application 302, possibly operated by a user, may provide an interface to the testing logic 220. The application 302 may utilize a basic input/output system (BIOS) call, a high-level driver, or any other means of accessing the testing logic 220 through a memory bus.

**[0019]** The application 302, possibly at the behest of a user, may send a request 304 to the testing logic 220 via a bus 224. The request 304 may represent one or more memory errors to be injected into the chipset. The injected

errors may be continuous, lasting indefinitely, or "single-shot," lasting for a single read or write cycle. Further, the errors may be correctable by the error correction code (ECC) implemented by the chipset or uncorrectable. In addition, the error may be injected into a specific location, such as a specific data line or specific memory region. The testing logic 220 may receive the request 304 and utilize the data stored in the SPD 218 memory and testing logic registers to formulate and inject an appropriate error injection procedure 306 onto one or more data lines of the memory module 200.

**[0020]** Referring now to Figure 4, an exemplary configuration of the testing logic 220 is shown in accordance with embodiments of the invention. The testing logic 220 may comprise one or more bus controllers that facilitate the transfer of data from the application 302 (Figure 3) and the SPD 218 memory to the testing logic 220. For example, the SPD 218 memory may be coupled to and operating on an inter-integrated circuit ($I^2C$) bus 404. The testing logic 220 may comprise an $I^2C$ controller 402 that facilitates the transfer of data to and from the $I^2C$ bus 404. The application 302 may communicate with the $I^2C$ bus 404 through the memory controller 104 (Figure 1) that may support the $I^2C$ bus 404. A serial data line (SDA) 406, a serial clock line (SCL) 408, and one or more serial address (SA) lines 410 may transfer data bi-directionally between the $I^2C$ controller 402 and the $I^2C$ bus 404. Thus, the testing logic 220 may receive input from the SPD 218 memory and the application 302 (Figure 3) via the $I^2C$ bus 404.

**[0021]** The testing logic 220 may also receive input via a chip select (CS) 412 line, a column address strobe (CAS) 416 line, and a row address strobe (RAS) 418 line. Other inputs to the testing logic 220, such as a write enable 414 line, a reset 402 line, and a clock 422 line, may be included as desired. The CS 412 line, the CAS 416 line, and the RAS 416 line may uniquely identify a unit of memory associated with the memory module 200. This unit of memory may represent a location in the memory module 200 at which to inject a memory error. The testing logic 220 may generate signals on one or more output 424 lines that may couple to one or more data lines of the memory module 200 to inject the memory error. Other logical components, such as QuickSwitches®, may be utilized as desired to assist in implementing the error injection procedure. In

addition, other storage components, such as registers, buffers, and queues, may be included in the testing logic 220 as desired.

**[0022]** Since the testing logic 220 comprises the $I^2C$ controller 402, the application 302 may address the testing logic 220 as a slave device on the $I^2C$ bus 404. Each slave device on an $I^2C$ bus is assigned a unique address that may be used to communicate with the device. Thus, the application 302 (Figure 3) may communicate with the testing logic 220 by sending the request 304 on the $I^2C$ bus 404, utilizing the slave address of the testing logic 220.

**[0023]** The testing logic 220 may perform various additional operations relating to memory test and validation. For example, the testing logic 220 may maintain one or more counters and inject a predetermined number of errors into the chipset. In addition, the testing logic 220 may be capable of testing an entire memory region or memory range for a particular memory error.

**[0024]** The above discussion is meant to be illustrative of the principles and various embodiments of the present invention. Numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.